UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/712,384	11/12/2003	William John Gallagher	BEAS-01316US3	9603

23910 7590 09/19/2006

FLIESLER MEYER, LLP
FOUR EMBARCADERO CENTER
SUITE 400
SAN FRANCISCO, CA 94111

EXAMINER	
NGUYEN, PHILLIP H	

ART UNIT	PAPER NUMBER
2193	

DATE MAILED: 09/19/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/712,384	GALLAGHER, WILLIAM JOHN	
	Examiner	Art Unit	
	Phillip H. Nguyen	2194	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *12 November 2003*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-22* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) _____ is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *12 November 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *20060421,20041012*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to the original filing of November 12, 2003. Claim 1-22 are pending and have been considered below.

Claim Objections

2. Claim 14 is objected to because of the following informalities: Claim 14 should depend on claim 1. Appropriate correction is required.

Specification

3. The amendment filed on August 15, 2006 overcomes the objection to paragraphs [0014-0015] of the previous office action. Therefore, the objection is withdrawn.

Claim Rejections - 35 USC § 112

4. The amendment filed on August 15, 2006 overcomes the rejection to claim 1, 8, and 9 that contain the trademark or trade name Java of the previous office action. Therefore, the rejection under 35 USC § 112 is withdrawn.

Double Patenting

5. A terminal disclaimer filed on August 15, 2006 overcomes the provisional rejection based on a nonstatutory double patenting over claims 1-11 of the previous copending office action. Therefore, the rejection is withdrawn.

Response to Amendment

6. The application (10/712,384) under 37 CFR 1.132 filed November 12, 2003 is insufficient to overcome the rejection of claims 1-11 based upon prior art (09/866,131) as set forth in the last Office action because:

Applicant argues Stapp fails to teach such recited claim limitation and therefore does not anticipate the embodiments of claim 1. Further Stapp does not suggest or otherwise render obvious the embodiments claimed by claim 1 either alone or in any combination since Stapp's approach teaches receiving rules from a user invocable programmer's tool. Modifications to Stapp to become a dynamic code generation instead of a manual tool would either (1) render Stapp inoperable or (2) require modifications to Stapp's purpose as well as Stapp's principle of operation (see MPEP 2143.01) because such modifications would necessarily burden Stapp's system contrary to their stated purpose: a user input data driven program generator (Abstract).

Examiner argument: Stapp's approach teaches receiving a set of data rules from a user using a first specification language to describe a desired computer program. The code generator will automatically generate the source code for whole new software applications, and for integrating newly generated source with existing project and environments (Paragraph 19). Stapp's approach might be manually setting up the organization and/or architecture, but once it receives all the required data rules from a user, it will automatically generate the source code. Therefore, Stapp's approach does not necessarily to modify in order to read on the application because at some initial point both the Stapp's approach and the application need to manually setting up the

required rules to satisfy the requirement before the generator can automatically generate the source code.

Claim Rejections - 35 USC § 102

7. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

8. Claim 1-9, 14-17, and 19-22 are rejected under 35 U.S.C. 102(b) as being anticipated by Stapp et al (09/866,131).

Claim 1: Stapp discloses a method for automatically generating program code comprising:

a. although Stapp does not explicitly disclose determining whether a resource is available. It is inherent in Stapp's approach that the resource must be available to the code generator in order to generate the source code (Figure 2, item 205);

b. creating a class file container object ([0024-0027]);

c. adding a method to the class file object ([0027]);

d. adding code to the method ([0027]); and

e. generating byte code for the class file object ([0036-0037]).

Claim 2: Stapp discloses a method for automatically generating program code as in claim 1 above, and further discloses wherein creating a class file container object includes: setting attributes for a class file ([0032]).

Claim 3: Stapp discloses the method in claim 2 above, and further discloses the attributes include at least one of class file name, parent super ([0033]).

Claim 4: Stapp discloses a method for automatically generating program code as in claim 1 above, and further discloses wherein adding a method to the class file object includes: adding a plurality of methods to the class file object ([0027]).

Claim 5: Stapp discloses a method for automatically generating program codes as in claim 1 above, and further discloses wherein adding code to the method includes adding code to the method using constructs that correspond to programming language statements, expressions, and variables ([0027], [0034]).

Claim 6: Stapp discloses the method in claim 5 above, and further discloses constructs include parameters ([0028]).

Claim 7: Stapp discloses the method in claim 5 above, and further discloses wherein each statement, expression type, variable is represented as an object ([0026-0027]).

Claim 8: Stapp discloses a method for automatically generating program code as in claim 1 above, and further discloses wherein generating byte code for the class file container object includes generating an intermediate representation of program flow (0037]).

Claim 9: Stapp discloses the method in claim 8 above, and further discloses wherein generating byte code for the class file container object includes converting the intermediate representation into byte code ([0037]).

Claim 14: Stapp discloses a method for automatically generating program code as in claim 1 above, and further discloses an object is a programming unit that groups together a data structure (one or more instance variables) and the operations (methods) that can use or affect that data ([0027]). Although, Stapp does not explicitly disclose repeatedly adding code for each method added to the class file. It is inherent that code must be repeatedly added to each method when needed in order for the method and the class file to perform correctly and for the code generator to fulfill its purpose.

Claim 15: Stapp discloses a method for automatically generating program code as in claim 1 above, and further discloses generating a tree of statement (a data structure (one or more instance variables) and the operations (methods)) ([00274]).

Claim 16: Stapp discloses a method for automatically generating program code as in claim 15 above, and further discloses generating a tree of statements includes generating a tree representing at least one method, the at least one method comprising at least one of a code statement, an expression, a variable and a programming construct ([0027]).

Claim 17: Stapp discloses the method as in claim 15 above, and further discloses generating a tree of statements includes generating a tree forming a known structure when the class file container is a known type ([0031-0032]).

Claim 19: Stapp discloses the method as in claim 1 above, and further discloses generating byte code for the class file container object includes maintaining a state of a program being generated by each statement ([0039]).

Claim 20: Stapp discloses the method as in claim 19 above, and further discloses maintaining at least one of a content of a stack, a content of a variable in use during program flow ([0039]).

Claim 21: Stapp discloses the method as in claim 20 above, and further discloses an object is a generic term that is used in the object oriented programming environment to refer to a module that contains related code and variables and also further discloses Java classes are compiled into byte code (intermediate

representation), which are executed by a machine-dependent virtual machine ([0037]). Although, Stapp does not explicitly disclose generating an intermediate representation of program flow based upon the at least one of a content of a stack, a content of a variable in use during program flow. It is inherent in Stapp's approach that when generating an intermediate representation of program flow, it must based upon the at least one of a content of a stack, a content of a variable in use during program flow in order to generate the intermediate representation.

Claim 22: Stapp discloses the method as in claim 1 above, and further discloses user interface component to assist users input and communicate data to the code generation component ([0071]). Stapp does not explicitly disclose determining whether a remote object having an interface to which code is being written is available. It is inherent that the interface is available to assist users input and communicate data to the code generation.

Claim Rejections - 35 USC § 103

9. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

10. Claim 10-13 and 18 are rejected under 35 U.S.C. 103(a) as being unpatentable over Stapp (09/866,131) in view of Goodwin (US 6,199,195).

Claim 10: Stapp discloses a method for automatically generating program code as in claim 1 above, but does not explicitly disclose that the program code implements an adaptor class. However, Goodwin discloses the program code implements an adapter class (Fig. 3, item 310, 312). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to add an adapter class in Stapp to take the place of some class and expose it through a different interface. One would have been motivated to use an adapter class when creating a reusable class that cooperates with other classes that have incompatible interfaces. Or when separating the client interface of an object from its implementation so each can evolve independently. Or when converting the interface of a class into another interfaces that the clients expect.

Claim 11: Stapp discloses a method for automatically generating program code as in claim 1 above, but does not explicitly disclose that the program code implement a proxy class. However, Goodwin discloses the program code implements a proxy class (Fig. 3, item 318). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to add a proxy class in Stapp to enforce call-by-value semantics between Enterprise JavaBeans (EJBs). One would have been motivated to use proxy class to reduce system development time, compile and build time, and system modification time.

Claim 12: Stapp discloses a method for automatically generating program code as in claim 1 above, and further discloses an object consists of data and one or more operations (methods) or procedures that can be performed on that data ([0027]). However, Stapp does not explicitly disclose repeatedly adding a method to the class file object for each method associated with a stub. However, it would have been obvious to one having an ordinary skill in the art at the time the invention was made to recognize that the repeatedly adding a method to the class file object for each method associated with a stub must occur in the Stapp's approach in order for an object to consist of data and one or more methods that can be performed on that data. Even if it is not occur in Stapp's approach, it still would have been obvious to one having an ordinary skill in the art at the time the invention was made to modify the code to repeatedly add a method to the class file object in order to fulfill the code generation purposes. Therefore, one would have been motivated to modify the code to add this step in Stapp's approach in order for the code generation to generate the source code.

Claim 13: Stapp discloses the method as in claim 12 above, but does not explicitly disclose determining a number of method s associated with the stub in a remote interface. However, it would have been obvious to one having an ordinary skill in the art at the time the invention was made to modify the code by including a counter or an indicator that increment each time a method is adding to the class that associated wit the stub. Therefore, one would have been motivated to add an indicator or a

counter to the code that increment each time a method is adding to the class object that associated with the stub so it would know when it reaches the end of adding iteration.

Claim 18: Stapp discloses the method as in claim 17 above, but does not explicitly disclose generating a tree forming a known structure when the class file container is of at least one of an adapter and a proxy type. Goodwin discloses the generating a tree forming a known structure when the class file container is of a proxy type (Col 7, line 50-60). It would have been obvious to one having an ordinary skill in the art at the time the invention was made to include the proxy class type in the Stapp's approach to enforce call-by-value semantics between Enterprise JavaBeans (EJBs). One would have been motivated to use proxy class to reduce system development time, compile and build time, and system modification time.

Conclusion

11. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of

the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

12. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

a. Taylor (US 2004/0019596 A1) discloses method, system, and program for making objects available for access to a client over a network.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Phillip H. Nguyen whose telephone number is (571) 270-1070. The examiner can normally be reached on Monday - Friday 10:00 AM - 3:00 PM EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, James Myhre can be reached on (571) 270-1065. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

P.N.
PN
9/13/06

James W. Myhre
Supervisory Primary Examiner